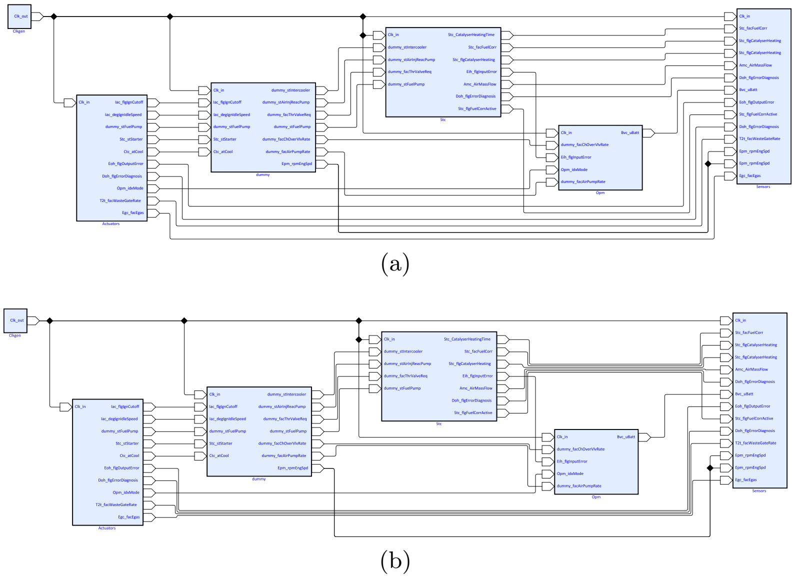
Edge Bundling for Dataflow Diagrams

Ulf Rüegg, Christoph Daniel Schulze, Carsten Sprung,
Nis Wechselberg, and Reinhard von Hanxleden

Dept. of Computer Science, Kiel University, Kiel, Germany
{uru,cds,csp,nbw,rvh}@informatik.uni-kiel.de

Edge bundling is a well-known technique to reduce visual clutter in node-link diagrams by having different links share the same path through the diagram [1,2,4,3]. *Dataflow diagrams* consist of functional blocks (*nodes*) that transfer data through channels (*links* or *edges*, usually routed orthogonally) that connect the blocks through dedicated connection points (*ports*). A natural concept of dataflow diagrams which is similar to edge bundling is the usage of *hyperedges* which can connect more than two nodes. Here we show how edge bundles can sensibly be incorporated into dataflow diagrams and how they compare to and can coexist with hyperedges. We briefly discuss methods to compute edge bundles as part of the *layer-based approach* to layout [7].

Edge Bundles vs. Hyperedges. Hyperedges are part of the diagram's structure and distribute the same data between the connected ports. Edge bundles on the other hand are a means of presentation and are formed by combining edges suitably. They abstract from port connections and instead emphasize which nodes



(a)

(b)

Fig. 1: (a) A dataflow diagram with a hyperedge between the top-left node and every other node. (b) The same diagram but with snuggling edge bundles.

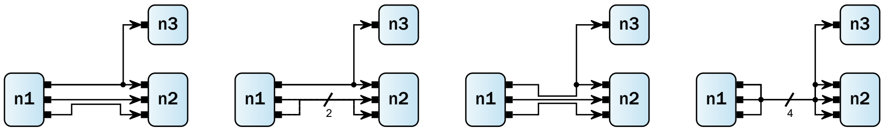


Fig. 2: Coexistence of edge bundles and hyperedges: four possible drawing styles.

are connected, which is meant to reduce visual clutter possibly losing the more detailed port connectivity information. In the context of dataflow diagrams, we want to restrict this loss by putting constraints on which edges can be bundled: two edges can only be bundled if they connect the same nodes. An edge bundle then illustrates that *some* data is exchanged between these nodes, with the exact data not being further specified.

Visual Representation. Hyperedges share as much of their path as possible and junction points are often emphasized using markers, e. g. little circles. As illustrated in Fig. 2, we propose four possibilities when drawing dataflow diagrams with edge bundles. We first distinguish between keeping hyperedges and edge bundles separate or combining them. Within each of these cases, we further distinguish whether to slightly separate the edges in each edge bundle (thus drawing them in a "snuggling" fashion), or to combine them. Separating hyperedges and edge bundles while drawing the bundles in a snuggling fashion retains the original connectivity information; the other drawing styles do not necessarily do so.

Methods. Given a finished drawing, we pursue two use cases: a) Edges are bundled without moving the nodes, which preserves the mental map of a user and allows regular and bundled edge routing in the same diagram. The user can interactively switch between the routing styles or "un-bundle" a single bundle to see the explicit connections. Nevertheless, care has to be taken not to produce unfortunate edge overlaps in the latter case. b) Node positions are allowed to be altered to produce smaller drawings by leveraging the space freed by combining edges.

The initial drawing can be computed using an existing layer-based method supporting ports and orthogonal edges [6]. An orthogonal edge consists of vertical and horizontal segments, the former of which are always placed in between *layers* and ordered to reduce edge crossings. For a bundle of edges a common route can directly be derived from the horizontal and vertical segments of the individual edges. A weighted shortest-path on an auxiliary graph determines the best suiting horizontal segments which induce the required height of the vertical segments. The order of (bundled) vertical segments between layers should be recomputed since the crossing number may change (we count a crossing with a bundle only once, even for snuggling bundles). Alternatively, a *constraint graph* can be formed from nodes and vertical segments and one-dimensional compaction techniques can be used to obtain a more compact drawing [5]. Appropriate implementations of the suggested methods are fast enough for interactive applications.

References

1. Eppstein, D., Goodrich, M.T., Meng, J.Y.: Confluent layered drawings. Algorithmica 47(4), 439–452 (2007)
2. Holten, D., van Wijk, J.J.: Force-directed edge bundling for graph visualization. Comput. Graph. Forum 28(3), 983–990 (2009)
3. Onoue, Y., Kukimoto, N., Sakamoto, N., Koyamada, K.: Minimizing the number of edges via edge concentration in dense layered graphs. IEEE Trans. Vis. Comput. Graph. 22(6), 1652–1661 (2016)
4. Pupyrev, S., Nachmanson, L., Kaufmann, M.: Improving layered graph layouts with edge bundling. In: Graph Drawing - 18th International Symposium, GD 2010, Konstanz, Germany, September 21-24, 2010. Revised Selected Papers. pp. 329–340 (2010)
5. Rüegg, U., Schulze, C.D., Grevismühl, D., von Hanxleden, R.: Using one-dimensional compaction for smaller graph drawings. In: Proceedings of the 9th International Conference on the Theory and Application of Diagrams (DIAGRAMS'16) (2016)
6. Schulze, C.D., Spönemann, M., von Hanxleden, R.: Drawing layered graphs with port constraints. Journal of Visual Languages and Computing, Special Issue on Diagram Aesthetics and Layout 25(2), 89–106 (2014)
7. Sugiyama, K., Tagawa, S., Toda, M.: Methods for visual understanding of hierarchical system structures. IEEE Transactions on Systems, Man and Cybernetics 11(2), 109–125 (Feb 1981)